

CLAIMS

- 1 1. A computer system configured to:
- 2 A) provide a task-queue set that includes at least one task queue in which can
- 3 be stored and from which can be retrieved task identifiers, which identify
- 4 tasks to be performed; and
- 5 B) for each task queue, employ a separate execution thread associated there-
- 6 with to:
- 7 i) select repeatedly between a LIFO access mode and a FIFO access
- 8 mode in accordance with a mode-selection criterion; and
- 9 ii) perform dynamically identified tasks by repeatedly:
- 10 a) popping a task identifier from the associated task queue in
- 11 accordance with an access mode thus selected;
- 12 b) so performing the task thereby identified as, in at least
- 13 some instances, to find one or more further tasks to be per-
- 14 formed; and
- 15 c) pushing onto the task queue task identifiers that identify
- 16 any tasks thus found.
- 1 2. A computer system as defined in claim 1 wherein pushing occurs at one, bottom
- 2 end of the queue, popping in accordance with the FIFO access mode occurs at the other,
- 3 top end of the queue, and popping in accordance with the LIFO access mode occurs at the
- 4 bottom end of the queue.
- 1 3. A computer system as defined in claim 1 wherein the queue accesses are circular.
- 1 4. A computer system as defined in claim 1 wherein the task-queue set includes of
- 2 plurality of the task queues.

- 1 5. A computer system as defined in claim 4 wherein each said dynamically identi-
2 fied task is the garbage-collection task of performing, for a given object associated with
3 that task, processing that includes identifying in the given object references to other ob-
4 jects and thereby identifying the tasks of performing similar processing for those other
5 objects.
- 1 6. A computer system as defined in claim 5 wherein the task identifiers are identifi-
2 ers of the objects associated with tasks that the task identifiers identify.
- 1 7. A computer system as defined in claim 6 wherein the task identifiers are pointers
2 to the objects associated with the tasks that the task identifiers identify.
- 1 8. A computer system as defined in claim 4 wherein, in at least some instances, an
2 execution thread associated with a task queue that is empty:
- 3 A) pops a task identifier from a task queue other than the one with which it is
4 associated;
- 5 B) so performs the task thereby identified as, in at least some instances, to
6 find one or more further tasks to be performed; and
- 7 C) pushes onto the task queue associated with it task identifiers that identify
8 any tasks thus found.
- 1 9. A computer system as defined in claim 8 wherein each said dynamically identi-
2 fied task is the garbage-collection task of performing, for a given object associated with
3 that task, processing that includes identifying in the given object references to other ob-
4 jects and thereby identifying the tasks of performing similar processing for those other
5 objects.
- 1 10. A compiler/interpreter that, in response to signals representing instructions that
2 define operations in which memory for data objects is allocated dynamically, generating

3 signals representing instructions that implement a garbage collector that operates in gar-
4 bage-collection cycles of which each includes an operation that includes:

- 5 A) providing a task-queue set that includes at least one task queue in which
6 can be stored and from which can be retrieved task identifiers, which
7 identify tasks to be performed; and
- 8 B) for each task queue, employing a separate execution thread associated
9 therewith to:
 - 10 i) select repeatedly between a LIFO access mode and a FIFO access
11 mode in accordance with a mode-selection criterion; and
 - 12 ii) perform dynamically identified tasks by repeatedly:
 - 13 a) popping a task identifier from the associated task queue in
14 accordance with an access mode thus selected;
 - 15 b) so performing the task thereby identified as, in at least
16 some instances, to find one or more further tasks to be per-
17 formed; and
 - 18 c) pushing onto the task queue task identifiers that identify
19 any tasks thus found.

1 11. A compiler/interpreter as defined in claim 10 wherein the task-queue set includes
2 of plurality of the task queues.

1 12. A compiler/interpreter as defined in claim 11 wherein, in at least some instances,
2 an execution thread associated with a task queue that is empty:

- 3 A) pops a task identifier from a task queue other than the one with which it is
4 associated;
- 5 B) so performs the task thereby identified as, in at least some instances, to
6 find one or more further tasks to be performed; and
- 7 C) pushes onto the task queue associated with it task identifiers that identify
8 any tasks thus found.

1 13. A compiler/interpreter as defined in claim 10 wherein the task identifiers are
2 identifiers of the objects associated with tasks that the task identifiers identify.

1 14. A compiler/interpreter as defined in claim 13 wherein the task identifiers are
2 pointers to the objects associated with the tasks that the task identifiers identify.

1 15. For performing dynamically identified tasks, a method comprising employing a
2 computer system to:

3 A) provide a task-queue set that includes at least one task queue in which can
4 be stored and from which can be retrieved task identifiers, which identify
5 tasks to be performed; and

6 B) for each task queue, employ a separate execution thread associated there-
7 with to:

8 i) select repeatedly between a LIFO access mode and a FIFO access
9 mode in accordance with a mode-selection criterion; and

10 ii) perform dynamically identified tasks by repeatedly:

11 a) popping a task identifier from the associated task queue in
12 accordance with an access mode thus selected;

13 b) so performing the task thereby identified as, in at least
14 some instances, to find one or more further tasks to be per-
15 formed; and

16 c) pushing onto the task queue task identifiers that identify
17 any tasks thus found.

1 16. A method as defined in claim 15 wherein pushing occurs at one, bottom end of
2 the queue, popping in accordance with the FIFO access mode occurs at the other, top end
3 of the queue, and popping in accordance with the LIFO access mode occurs at the bottom
4 end of the queue.

1 17. A method as defined in claim 15 wherein the queue accesses are circular.

1 18. A method as defined in claim 15 wherein the task-queue set includes of plurality
2 of the task queues.

1 19. A method as defined in claim 18 wherein each said dynamically identified task is
2 the garbage-collection task of performing, for a given object associated with that task,
3 processing that includes identifying in the given object references to other objects and
4 thereby identifying the tasks of performing similar processing for those other objects.

1 20. A method as defined in claim 19 wherein the task identifiers are identifiers of the
2 objects associated with tasks that the task identifiers identify.

1 21. A method as defined in claim 20 wherein the task identifiers are pointers to the
2 objects associated with the tasks that the task identifiers identify.

1 22. A method as defined in claim 18 wherein, in at least some instances, an execution
2 thread associated with a task queue that is empty:

3 A) pops a task identifier from a task queue other than the one with which it is
4 associated;

5 B) so performs the task thereby identified as, in at least some instances, to
6 find one or more further tasks to be performed; and

7 C) pushes onto the task queue associated with it task identifiers that identify
8 any tasks thus found.

1 23. A method as defined in claim 22 wherein each said dynamically identified task is
2 the garbage-collection task of performing, for a given object associated with that task,
3 processing that includes identifying in the given object references to other objects and
4 thereby identifying the tasks of performing similar processing for those other objects.

1 24. A storage medium containing instructions readable by a computer system to cause
2 the computer system to:

3 A) provide a task-queue set that includes at least one task queue in which can
4 be stored and from which can be retrieved task identifiers, which identify
5 tasks to be performed; and

6 B) for each task queue, employ a separate execution thread associated there-
7 with to:

8 i) select repeatedly between a LIFO access mode and a FIFO access
9 mode in accordance with a mode-selection criterion; and

10 ii) perform dynamically identified tasks by repeatedly:

11 a) popping a task identifier from the associated task queue in
12 accordance with an access mode thus selected;

13 b) so performing the task thereby identified as, in at least
14 some instances, to find one or more further tasks to be per-
15 formed; and

16 c) pushing onto the task queue task identifiers that identify
17 any tasks thus found.

1 25. A storage medium as defined in claim 24 wherein pushing occurs at one, bottom
2 end of the queue, popping in accordance with the FIFO access mode occurs at the other,
3 top end of the queue, and popping in accordance with the LIFO access mode occurs at the
4 bottom end of the queue.

1 26. A storage medium as defined in claim 24 wherein the queue accesses are circular.

1 27. A storage medium as defined in claim 24 wherein the task-queue set includes of
2 plurality of the task queues.

1 28. A storage medium as defined in claim 27 wherein each said dynamically identi-
2 fied task is the garbage-collection task of performing, for a given object associated with

3 that task, processing that includes identifying in the given object references to other ob-
4 jects and thereby identifying the tasks of performing similar processing for those other
5 objects.

1 29. A storage medium as defined in claim 28 wherein the task identifiers are identifi-
2 ers of the objects associated with tasks that the task identifiers identify.

1 30. A storage medium as defined in claim 29 wherein the task identifiers are pointers
2 to the objects associated with the tasks that the task identifiers identify.

1 31. A storage medium as defined in claim 27 wherein, in at least some instances, an
2 execution thread associated with a task queue that is empty:
3 A) pops a task identifier from a task queue other than the one with which it is
4 associated;
5 B) so performs the task thereby identified as, in at least some instances, to
6 find one or more further tasks to be performed; and
7 C) pushes onto the task queue associated with it task identifiers that identify
8 any tasks thus found.

1 32. A storage medium as defined in claim 31 wherein each said dynamically identi-
2 fied task is the garbage-collection task of performing, for a given object associated with
3 that task, processing that includes identifying in the given object references to other ob-
4 jects and thereby identifying the tasks of performing similar processing for those other
5 objects.

1 33. A signal representing a sequence of instructions that, when they are executed by
2 computer system, cause the computer system to:
3 A) provide a task-queue set that includes at least one task queue in which can
4 be stored and from which can be retrieved task identifiers, which identify
5 tasks to be performed; and

- 6 B) for each task queue, employ a separate execution thread associated there-
7 with to:
8 i) select repeatedly between a LIFO access mode and a FIFO access
9 mode in accordance with a mode-selection criterion; and
10 ii) perform dynamically identified tasks by repeatedly:
11 a) popping a task identifier from the associated task queue in
12 accordance with an access mode thus selected;
13 b) so performing the task thereby identified as, in at least
14 some instances, to find one or more further tasks to be per-
15 formed; and
16 c) pushing onto the task queue task identifiers that identify
17 any tasks thus found.

1 34. A signal as defined in claim 33 wherein pushing occurs at one, bottom end of the
2 queue, popping in accordance with the FIFO access mode occurs at the other, top end of
3 the queue, and popping in accordance with the LIFO access mode occurs at the bottom
4 end of the queue.

1 35. A signal as defined in claim 33 wherein the queue accesses are circular.

1 36. A signal as defined in claim 33 wherein the task-queue set includes of plurality of
2 the task queues.

1 37. A signal as defined in claim 36 wherein each said dynamically identified task is
2 the garbage-collection task of performing, for a given object associated with that task,
3 processing that includes identifying in the given object references to other objects and
4 thereby identifying the tasks of performing similar processing for those other objects.

1 38. A signal as defined in claim 37 wherein the task identifiers are identifiers of the
2 objects associated with tasks that the task identifiers identify.

1 39. A signal as defined in claim 38 wherein the task identifiers are pointers to the ob-
2 jects associated with the tasks that the task identifiers identify.

1 40. A signal as defined in claim 36 wherein, in at least some instances, an execution
2 thread associated with a task queue that is empty:

- 3 A) pops a task identifier from a task queue other than the one with which it is
4 associated;
5 B) so performs the task thereby identified as, in at least some instances, to
6 find one or more further tasks to be performed; and
7 C) pushes onto the task queue associated with it task identifiers that identify
8 any tasks thus found.

1 41. A signal as defined in claim 40 wherein each said dynamically identified task is
2 the garbage-collection task of performing, for a given object associated with that task,
3 processing that includes identifying in the given object references to other objects and
4 thereby identifying the tasks of performing similar processing for those other objects.

1 42. A computer system comprising:

- 2 A) means for providing a task-queue set that includes at least one task queue
3 in which can be stored and from which can be retrieved task identifiers,
4 which identify tasks to be performed; and
5 B) for each task queue, means for employing a separate execution thread as-
6 sociated therewith to:
7 i) select repeatedly between a LIFO access mode and a FIFO access
8 mode in accordance with a mode-selection criterion; and
9 ii) perform dynamically identified tasks by repeatedly:
10 a) popping a task identifier from the associated task queue in
11 accordance with an access mode thus selected;

- 12 b) so performing the task thereby identified as, in at least
13 some instances, to find one or more further tasks to be per-
14 formed; and
15 c) pushing onto the task queue task identifiers that identify
16 any tasks thus found.

112047-0041.P5388